AMENDMENTS TO THE CLAIMS

The following listing of claims will replace all prior versions and listing of claims in the application.

Listing of Claims:

1. (Original) A system for managing communication between processes, comprising:

a set of memory blocks, the set of memory blocks being accessible by a plurality of processes each having at least one assigned memory block in the set of memory blocks; and

a communications engine, the communications engine communicating with the set of memory blocks, the communications engine directing a data access request from at least one of the plurality of processes to the at least one corresponding assigned memory block based on memory management data.

2. (Original) A system according to claim 1, further comprising an administrative memory, the administrative memory being separate from the shared memory space.

3. (Original) A system according to claim 2, wherein the memory management data is stored at least in the administrative memory.

4. (Original) A system according to claim 3, wherein the memory management data comprises a plurality of memory management objects associated with the plurality of processes.

5. (Original) A system according to claim 4, wherein each memory management object is uniquely associated with one of the plurality of processes.

6. (Original) A system according to claim 4, wherein the plurality of memory management objects are instantiated from classes generated by the communications engine.

7. (Original) A system according to claim 1, wherein the set of memory blocks comprises a first memory address space.

8. (Original) A system according to claim 7, wherein the administrative memory comprises a second memory address space.

9. (Original) A system according to claim 2, wherein the each of the processes register their at least one assigned memory block in the set of memory blocks in the administrative memory.

10. (Original) A system according to claim 1, wherein the data access request comprises a modification to the data in the at least one assigned memory block.

11. (Original) A system according to claim 1, wherein the data access request returns data from the at least one assigned memory block.

12. (Original) A system according to claim 11, wherein the communications engine parses at least one of the data access request and the returned data.

13. (Original) A system according to claim 1, wherein the communications engine comprises a cache module, the cache module maintaining a cache of data exchanged between the set of memory blocks.

14. (Original) A system according to claim 1, wherein the communications engine assigns the at least one memory block to each process based on a buddy system allocation.

15. (Original) A system according to claim 1, wherein the communications engine terminates a connection to the set of memory blocks after completion of the data access request.

16. (Original) A system according to claim 1, wherein the at least one assigned memory block is dynamically assigned to each process.

17. (Original) A system according to claim 1, wherein each process further comprises a process interface to an application programming interface.

18. (Original) A system according to claim 1, wherein the communications engine further comprises an engine interface to an application programming interface.

19. (Original) A system according to claim 1, wherein each memory block in the set of memory blocks is accessible only by at least one of the communications engine and its corresponding process.

20. (Original) A system according to claim 1, wherein a process of communication between two processes comprises copying data in one memory block to another memory block.

21. (Original) A system according to claim 1, wherein a process of communication between two processes comprises transmitting data from one memory block to a remote destination process.

22. (Original) A system according to claim 21, wherein the transmitted data is transmitted via an external network to the destination process.

23. (Original) A system according to claim 1, wherein the communications engine is separate from an operating system supporting the plurality of processes.

24. (Original) A method for managing communication between processes, comprising:

assigning at least one memory block from a set of memory blocks to each of a plurality of processes; and

directing a data access request from at least one of the plurality of processes to the at least one corresponding assigned memory block based on memory management data.

25. (Original) A method according to claim 24, further comprising a step of storing the memory management data in at least an administrative memory, the administrative memory being separate from the shared memory space.

26. (Original) A method according to claim 24, wherein the memory management data comprises a plurality of memory management objects associated with the plurality of processes.

27. (Original) A method according to claim 26, wherein each memory management object is uniquely associated with one of the plurality of processes.

28. (Original) A method according to claim 26, further comprising a step of instantiating the plurality of memory management objects from classes generated by the communications engine.

29. (Original) A method system according to claim 24, wherein the set of memory blocks comprises a first memory address space.

30. (Original) A method according to claim 29, wherein the administrative memory comprises a second memory address space.

31. (Original) A method according to claim 25, further comprising a step of each processes registering their at least one assigned memory block in the set of memory blocks in the administrative memory.

32. (Original) A method according to claim 24, further comprising a step of modifying the data in at least one assigned memory block according to the data access request.

33. (Original) A method according to claim 24, further comprising a step of returning data from at least one assigned memory block according to the data access request.

34. (Original) A method according to claim 33, further comprising a step of parsing at least one of the data access request and the returned data.

35. (Original) A method according to claim 24, further comprising a step of executing a cache module, the cache module maintaining a cache of data exchanged between the set of memory blocks.

36. (Original) A method according to claim 24, further comprising a step of assigning the at least one memory block to each process based on a buddy system allocation.

37. (Original) A method according to claim 24, further comprising a step of terminating a connection to the set of memory blocks after completion of the data access request.

38. (Original) A method according to claim 24, wherein the step of assigning at least one memory block to each of the plurality of processes comprises a step of dynamically assigning at least one memory to each of the plurality of processes.

39. (Original) A method according to claim 24, further comprising a step of presenting an application programming interface to at least one of the plurality of processes and the communications engine.

40. (Original) A method according to claim 24, further comprising a step of limiting access to each memory block in the set of memory blocks to at least one of the

communications engine and its corresponding process.

41. (Original) A method according to claim 24, further comprising a step of copying data in one memory block to another memory block.

42. (Original) A method according to claim 24, further comprising a step of transmitting data from one memory block to a remote destination process.

43. (Original) A method according to claim 42, wherein the step of transmitting data comprises a step of transmitting data via an external network to the destination process.

44. (Original) A method according to claim 24, wherein the communications engine is separate from an operating system supporting the plurality of processes.

45. (Original) A system for managing communication between processes, comprising:

memory block means, the memory block means being accessible by a plurality of processes each having at least one assigned memory block in the memory block means; and

communications engine means, the communications engine means communicating with the memory block means to direct a data access request from at least one of the plurality of processes to the at least one corresponding assigned memory block based on memory management data.

46. (Original) A system according to claim 1, further comprising administrative memory means, the administrative memory means being separate from the shared memory space.

47. (Original) A system according to claim 46, wherein the memory management data is stored at least in the administrative memory means.

48. (Original) A system according to claim 45, further comprising application

programming interface means, the application programming interface means being callable by at least one of the plurality of processes and the communications engine means to perform defined tasks.

49. (Original) A system according to claim 45, wherein the communications engine means is separate from an operating system supporting the plurality of processes.

50. (Original) A computer readable medium, the computer readable medium being readable to execute a method for managing communication between processes, the method comprising:

assigning at least one memory block from a set of memory blocks to each of a plurality of processes; and

directing a data access request from at least one of the plurality of processes to the at least one corresponding assigned memory block based on memory management data.

51. (Original) A computer readable medium according to claim 50, wherein the method further comprises a step of storing the memory management data in at least an administrative memory, the administrative memory being separate from the shared memory space.

52. (Original) A computer readable medium according to claim 50, wherein the step of assigning at least one memory block to each of the plurality of processes comprises a step of dynamically assigning at least one memory to each of the plurality of processes.

53. (Original) A computer readable medium according to claim 50, wherein the method further comprises a step of presenting an application programming interface to at least one of the plurality of processes and the communications engine.

54. (Original) A computer readable medium according to claim 50, wherein the method

further comprises a step of limiting access to each memory block in the set of memory blocks to at least one of the communications engine and its corresponding process.

55. (Original) A computer readable medium according to claim 50, wherein the method further comprises a step of copying data in one memory block to another memory block.

56. (Original) A computer readable medium according to claim 50, wherein the method further comprises a step of transmitting data from one memory block to a remote destination process.

57. (Original) A computer-executable process, the computer-executable process being configured to communicate with other processes according to a method of:

registering a request to receive an assignment of at least one memory block from a set of memory blocks in a shared memory space shared by a plurality of processes;

receiving at least one assigned memory block in the shared memory space; and

processing a data access request for data to or from at least one of the plurality of processes via a communications engine, the communications engine directing the data access request engine to a corresponding assigned memory block based on memory management data.

58. (Original) A computer-executable process according to claim 57, wherein the method further comprises a step of storing the memory management data in at least an administrative memory, the administrative memory being separate from the shared memory space.

59. (Original) A computer-executable process according to claim 57, wherein the method further comprises a step of invoking a plurality of memory management objects

associated with the plurality of processes.

60. (Original) A computer-executable process according to claim 57, wherein the method further comprises a step of invoking an application programming interface.

61. (Original) A computer-executable process according to claim 57, wherein the method further comprises a step of copying data in one memory block to another memory block.

Claims 62-66 (Cancelled).